Cornell University
CS 5788: Introduction to Generative Models
Spring 2026 Instructor: Andrew Owens

Problem Set 1: Simple generative models

Posted: Friday, January 23, 2026	**Due:** Tuesday, February 10, 2026

Please submit your written solution to Gradescope as a `.pdf` file. Please convert your Colab notebooks to PDF. For your convenience, we have included the PDF conversion script at the end of the notebook.

The starter code was updated on Jan. 30. Please see Ed Discussion for details.

Starter code:

- The notebook for the programming problem can be found at:
  https://drive.google.com/file/d/1Xy_-GZ_ekkczrvRVZOLyxnzknh2FiEVm/view?usp=sharing
- Submit your solution to the written problems as a PDF file (either written or typeset).

We recommend editing and running your code in Google Colab, although you are welcome to use your local machine instead. Please note that problems marked optional will not be graded.

Problem 1.0 `numpy` *and PyTorch review (optional)*

We have provided Colab notebooks containing brief reviews of `numpy` here and PyTorch here.

Problem 1.1 *Gaussian mixture models for image patches*

We will use a Gaussian mixture model (GMM) to learn the distribution of tiny 16×16 pixel image patches.

(a) We will start by fitting a simple multivariate Gaussian distribution to image patches:

$$p_\theta(\mathbf{x}) = \frac{1}{\sqrt{(2\pi)^d \det(\mathbf{\Sigma})}} \exp\left(-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^\top \mathbf{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})\right), \tag{1}$$

where d is the dimension of the (flattened) image patch. Estimate $\boldsymbol{\mu}$ and $\mathbf{\Sigma}$ from image patches using the closed form solutions discussed in class.

(b) As described in class, we can write the process of sampling from a Gaussian as:

$$\mathbf{x} = \mathbf{V}\mathbf{D}^{\frac{1}{2}}\mathbf{z}, \tag{2}$$

where $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ is sampled from an i.i.d. Gaussian, $\mathbf{D}$ is a diagonal matrix of eigenvalues, and $\mathbf{V}$ is the matrix of column eigenvectors. One can therefore interpret the $d \times d$ covariance matrix $\mathbf{\Sigma} = \mathbf{V}\mathbf{D}\mathbf{V}^\top$ as a dictionary containing d dictionary elements that each point to directions.

Given this interpretation, visualize the top 10 eigenvectors of the covariance matrix. Describe their appearance in words.

(c) (Optional) We have provided code to visualize $\text{Cov}[P_{p^*}, P_{q_i}]$, the covariance between the intensity value for a given pixel p^* and the intensity of all pixels q_i in the patch. (i) What does the structure of the covariance matrix suggest translational invariance of the input signal? (ii) Show that if $\mathbf{\Sigma}_{ij} = f(i-j)$ for some function f then the complex exponentials $\mathbf{v}[u] = \exp(iwu)$ for a constant w are eigenvectors of $\mathbf{\Sigma}$. (iii) Does this explain the appearance of the eigenvectors?

(d) Instead of using a closed form solution to estimate $\boldsymbol{\mu}$ and $\mathbf{\Sigma}$, we will use stochastic gradient descent (SGD). We have provided you with a partial implementation in PyTorch, which you should complete.

In your implementation:

- Minimize the negative log likelihood (NLL):

$$\mathbb{E}_{\mathbf{x}}\left[-\log(p_\theta(\mathbf{x}))\right] \approx -\frac{1}{N}\sum_{i=1}^{N}\log p_\theta(\mathbf{x}_i) \tag{3}$$

  for the dataset $\{\mathbf{x}_i\}_{i=1}^N$ and the probability density defined in Equation 1.
- Parameterize the covariance $\mathbf{\Sigma}$ using a matrix $\mathbf{A}$, such that $\mathbf{\Sigma} = \mathbf{A}\mathbf{A}^\top + \epsilon I$ using a constant $\epsilon = 10^{-4}$. This helps ensure that $\mathbf{\Sigma}$ is positive semidefinite.
- We recommend training the model using stochastic gradient descent for 20 epochs using a batch size of 512 or higher. We also recommend using the Adam optimizer with a learning rate of 10^{-4}.

Run the model on the test set and plot the NLL. Compare the final NLL to that of the closed form solution.

(e) Extend your approach by fitting a *Gaussian mixture model* (GMM), rather than a multivariate Gaussian. As described in class, the probability density for this model is:

$$q_\phi(\mathbf{x}) = \sum_{i=1}^{K}\pi_i p_{\theta_i}(\mathbf{x}), \tag{4}$$

where $\pi_1, \pi_2, ..., \pi_K$ are the mixture weights of each of the K components, the parameters $\phi = \{\theta_i\}_{i=1}^K = \{(\mu_i, \mathbf{\Sigma}_i)\}_{i=1}^K$ are the mean and covariance of the Gaussian distributions in the mixture, and p_{θ_i} is the density of a multivariate Gaussian (Eq. 1) with its given parameters.

To ensure that the mixture weights sum to 1, represent their *logits*, $\mathbf{w} \in \mathbb{R}^K$, computing the weights $\boldsymbol{\pi} = \text{softmax}(\mathbf{w})$:

$$\pi_i = \frac{\exp(w_i)}{\sum_{j=1}^{K}\exp(w_j)}. \tag{5}$$

Run your model using $K = 16$.

(f) Compare the NLL of the GMM and the plain Gaussian models on the test set. Which one has lower loss? Why?

(g) Fit a model with $K = 1, 4, 16, 32, 64$ mixture components. For each one, sample image patches from the model and plot the NLL. How does performance change, both quantitatively and qualitatively?

(h) Visualize the top eigenvectors of $\mathbf{\Sigma}_i$ for the $K = 64$ model using the provided code. For each mixture component i, the provided code shows the image patches for which $\text{argmax}_j \, p_{\theta_j}(\mathbf{x}) = i$. Find a component from the set of K that captures each of the following image structures: (1) edges, (2) "flat" regions of uniform intensity, (3) some other type of image content, of your choice (please describe what you see).

Problem 1.2 *Image denoising using a Gaussian mixture model*

The generative model that you learned in Problem 1.1 has a number of applications. We will use it to solve the *image denoising* problem — an important problem that we will encounter many times throughout the course.

We are given a noisy input image $\mathbf{X} \sim \mathcal{N}(\mathbf{X}_0, \sigma^2 I)$ that was obtained by corrupting a clean image $\mathbf{X}_0$ by Gaussian noise with standard deviation $\sigma = 25/255$. Our goal is to recover the clean image $\mathbf{X}_0$ from this noisy observation. Specifically, we will use our GMM as an *image prior* that distinguishes between realistic and unrealistic image patches.

We will estimate an image $\mathbf{X}$ that trades off between its similarity to the observed image and the likelihood of its patches under the GMM:

$$\mathcal{L}(\hat{\mathbf{X}}) = \|\hat{\mathbf{X}} - \mathbf{X}\|^2 - \lambda L_\theta(\hat{\mathbf{X}}), \tag{6}$$

where $\lambda = 1$ is a weight factor that controls the importance of the two terms and L_θ is the *expected patch log likelihood*. We define it to be:

$$L_\theta(\mathbf{X}) = \frac{1}{|\mathcal{P}(\mathbf{X})|} \sum_{\mathbf{x} \in \mathcal{P}(\mathbf{X})} \log q_\phi(\mathbf{x}), \tag{7}$$

where $\mathcal{P}(\mathbf{X})$ is the set of all (overlapping) image patches in the image and q_ϕ is your learned GMM model (Eq. 4).

(a) To start, we will denoise individual image patches rather than whole images. Implement this denoising approach by using gradient descent to solve for $\hat{\mathbf{x}}$. Then use the provided code to measure the mean squared prediction error (MSE), $\|\mathbf{x_0} - \hat{\mathbf{x}}\|^2$. Compare the accuracy of your model on the test set for $K = 1$ versus $K = 64$.

(b) Apply the approach to denoising a full image. Compare the qualitative results using $K = 1$ versus $K = 64$.

Problem 1.3 *Probability foundations*

(a) In class, we derived the closed form solution for $\boldsymbol{\mu}$, the mean of the multivariate Gaussian (Eq. 1). Do the same for $\boldsymbol{\Sigma}$, after making the assumption that it is diagonal:

$$\boldsymbol{\Sigma} = \text{diag}(\sigma_1^2, \sigma_2^2, ..., \sigma_d^2) = \begin{bmatrix} \sigma_1^2 & 0 & \cdots & 0 \\ 0 & \sigma_2^2 & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & \sigma_d^2 \end{bmatrix}$$

(b) Derive the mean of a single-variable Gaussian mixture model (Eq. 4).

(c) Suppose that $\mathbf{X} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$. Use the *change of variables* formula to show that the *whitened* variable $\mathbf{Z} = \boldsymbol{\Sigma}^{-\frac{1}{2}}(\mathbf{x} - \mu)$ is distributed as $\mathbf{Z} \sim \mathcal{N}(\mathbf{0}, I)$. Recall that $A^{\frac{1}{2}}$ is a *matrix square root* $A = A^{\frac{1}{2}}A^{\frac{1}{2}}$ and $A^{-\frac{1}{2}}$ is its inverse.

Hint: $\det(AB) = \det(A)\det(B)$.

(d) Show that:

$$D_{KL}\left(\mathcal{N}(0,1) \parallel \mathcal{N}(\mu, \sigma)\right) = \log(\sigma) + \frac{1+\mu^2}{2\sigma^2} - \frac{1}{2}.$$

Recall that KL divergence is defined as:

$$D_{KL}(p(x) \parallel q(x)) = \int_x p(x) \log\left(\frac{p(x)}{q(x)}\right) dx$$

for a where $\mathcal{N}(\mu, \sigma)$ is a single-variable Gaussian.

Hint: You should not need to evaluate any integrals! Instead, take advantage of the fact that for a Gaussian random variable $X \sim \mathcal{N}(\mu', \sigma')$ that $\mathbb{E}[X] = \mu'$ and $\mathbb{E}[(X - \mu')^2] = \sigma'$.

Credits. This problem set was written by Adnan Armouti, Jeongsoo Park, and Andrew Owens. It is based on the following papers:

[1] Daniel Zoran and Yair Weiss. From learning models of natural image patches to whole image restoration. In *2011 international conference on computer vision*, 2011.

[2] Daniel Zoran and Yair Weiss. Natural images, gaussian mixtures and dead leaves. *Advances in Neural Information Processing Systems*, 2012.